React - Perusteet

Tämän oppimateriaalin tavoitteena on tarjota pelkistetysti React-ohjelmoinnin ytimessä olevia asioita React-oppimispolun alussa olevalle. Tämä tavoite mielessä tähän materiaaliin ei ole otettu mukaan haitallista ylimäärää ammattimaiseen React-ohjelmointiin kuuluvia kehitystyötä tukevia käytänteitä ja työkaluja.

> "Learning React by copying boilerplates is like learning to cook by eating food in fancy restaurants. It doesn't work. You need to start with basics and ignore the fear of missing out."—Dan Abramov

Yleistä

- React on Facebookin ja Instagramin kehittäjien toteuttama JavaScript-kirjasto web-käyttöliittymien tekoon
- Käyttöliittymä toteutetaan luomalla omia React-komponentteja, jotka osaavat automaattisesti renderöidä/piirtää uuden tiedon näkyville
- React pitää taustalla sisäistä virtuaalista DOM-puuta, jonka kautta muutetaan oikeaan DOM-puuhun vain tarvittavat/halutut osat/solmut == nopeus
- Reactia ei pidetä varsinaisena frameworkkina, vaan MVC-mallin V:nä eli View:nä (kirjasto, joka renderöi näkymän)

Keskeistä:

- Pilkotaan yksittäinen sovellus komponenttien avulla niin pieniin osiin *"kuin on järkevää"* => helposti hallittavaa, ylläpidettävää ja testattavaa

Reactin käyttöönotto

Ohjelmoidaksesi JavaScriptillä Reactia käyttäviä sovelluksia tarvittavat React-kirjastot on osoitettava käytettäväksi tavalla tai toisella. Tähän on tarjolla useita vaihtoehtoja.

1. CDN-palvelut

Netistä löytyy useita CDN-palveluita, joista Reactin tarvittamat kirjastot voidaan helposti liittää oman HTML-tiedoston HEAD-elementtiin seuraavalla tavalla:

```
1  <!-- Main library-->
2  <script src="https://unpkg.com/react@16/umd/react.develo
3  <!-- DOM -->
4  <script src="https://unpkg.com/react-dom@16/umd/react-do
5  <!-- JSX -->
6  <script src="https://unpkg.com/babel-standalone@6.15.0/b
```

2. CodePen

CodePen-palveluun on toteutettu yksinkertainen Hello World -esimerkki, jonka avulla on helppo kokeilla Reactia. Esimerkin voi kopioida itselleen ja editoida sitä opettelutarkoituksessa.

Kokeile esimerkkiä täällä: Hello World -esimerkki

3. npm

React on tarjolla myös npm-ohjelman (Node Package Manager) kautta. Nodejs:n npm on ohjelmistopakettien hallintatyökalu, jonka avulla kehittäjät voivat etsiä, jakaa ja käyttää uudelleen muitakin kuin Nodejs:lle tarkoitettuja JavaScript-ohjelmointipaketteja. Järjestelmässäsi tulee tällöin olla asennettuna: Node.js (ja sen mukana asentuva npm.)

Tällöin React voidaan ottaa käyttöön GitHubista löytyvän Create React App -repositorion ja sen ohjeiden avulla.

Hello World! Reactilla

- Käytetään CDN-lähteitä => riittää ladata selaimeen vain tämä tiedosto
- React-sovelluksessa tarvitaan minimissään
  - Tarvittavat React-kirjastot, tässä rivit 7-12
  - Nimetty DIV-elementti, johon Reactin juurikomponentti kiinnitetään (rivi 16)
  - Reactin juurikomponentin ReactDOM.render()-metodi (rivit 19-22), jonka ensimmäinen argumentti (rivi 20) määrittää kiinnitettävän React-

komponentin tai HTML-elementin (tässä H1-elementti). Toinen argumentti (rivi 21) määrittää sen paikan, mihin tämä Reactin juurikomponentti HTML-osiossa sijoitetaan renderöitäväksi.

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="UTF-8" />
5      <title>Hello World</title>
6
7      <!-- Main library-->
8      <script src="https://unpkg.com/react@16/umd/react.de
9      <!-- DOM -->
10     <script src="https://unpkg.com/react-dom@16/umd/reac
11     <!-- JSX -->
12     <script src="https://unpkg.com/babel-standalone@6.15
13   </head>
14
15   <body>
16     <div id="root"></div>
17     <script type="text/babel">
18
19       ReactDOM.render(
20         <h1>Hello, world!</h1>,
21         document.getElementById('root')
22       );
23
24     </script>
25   </body>
26 </html>
```

Kokeile käytännössä: e/09/react01-99.html

React-sovelluksen keskeisiä käsitteitä

Komponentit

- Reactissa sovellus koostuu komponenteista.

- Komponenttien avulla toteutettava sovellus jaetaan pienempiin osiin, jotta sitä on helpompi hallita, kehittää, jne.
- Komponenttien render-metodissa määritellään komponentin ulkoasu. Määrittely voi sisältää
  - HTML-elementtejä,
  - omia Reactin komponentteja tai/ja
  - JavaScript-ohjelmointia.
- HTML-elementit kirjoitetaan pienillä kirjaimilla ja React-komponentit **aloitetaan** isolla kirjaimella.

Esimerkki react02-99.html

Havainnot

- Riveillä 16-25 on itse toteutettu Message-komponentti.
- ReactDOM.render-metodin kutsun ensimmäisenä argumenttina (rivi 29) välitetään määritelty Message-komponentti. Huomaa merkintätapa
- render-metodin tulee palauttaa yksi juurisolmu. Palautettaessa useita solmuja ne voidaan kääriä esim DIV-elementtiin.

```
1  <!doctype html>
2  <html>
3      <head>
4          <meta charset="utf-8">
5          <title>react02-99.html</title>
6      <script src="https://unpkg.com/react@16/umd/react.de
7      <script src="https://unpkg.com/react-dom@16/umd/reac
8      <script src="https://unpkg.com/babel-standalone@6.15
9
10     <body>
11
12     <div id="root"></div>
13
14     <script type="text/babel">
15
16 class Message extends React.Component {
17   render() {
18     return (
19       <div>
20         <h1>My Message is</h1>
```

```
            <p>Hello World!</p>
          </div>
        );
      }
    }

    ReactDOM.render(
        <Message />,
        document.getElementById("root")
    );

        </script>
    </body>
    </html>
```

Kokeile: e/09/react02-99.html

Esimerkki react03-99.html

Havainnot

- Kaksi luotua komponenttia: Message ja Messages.
- Messages-komponentti välittää dataa Message-komponentille
- Message-komponentti palauttaa p-elementin sisällä sille attribuutina annetun ja props-objektiin liitetyn sisällön
- Komponenttia voidaan käyttää useasti samassa React-toteutuksessa.

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>react03-99.html</title>
    <script src="https://unpkg.com/react@16/umd/react.de
    <script src="https://unpkg.com/react-dom@16/umd/reac
    <script src="https://unpkg.com/babel-standalone@6.15

    <body>

    <div id="root"></div>
```

```
13
14      <script type="text/babel">
15
16  class Messages extends React.Component {
17    render() {
18      return (
19        <div>
20          <h1>My Messages</h1>
21          <Message message="Message text here..." />
22          <Message message="Another message text here..."
23        </div>
24      );
25    }
26  }
27
28  class Message extends React.Component {
29    render() {
30      return (
31          <p>{this.props.message}</p>
32      );
33    }
34  }
35
36  ReactDOM.render(
37      <Messages />,
38      document.getElementById("root")
39  );
40
41      </script>
42  </body>
43  </html>
```

Kokeile: e/09/react03-99.html

JSX

- Komponenttien luonnissa käytetään yleisesti JSX-syntaksia
- JSX muistuttaa ulkonäöltään HTML/XML:ää -> visuaalisesti helpompi ymmärtää

- React-ohjelmointi on mahdollista puhtaalla JavaScriptillä ilman JSX:ää, mutta käytännössä se on harvinaista.
- JSX täytyy kääntää JavaScriptiksi -> Toteutetaan selaimessa automaattisesti sisällyttämällä babel-kirjasto sovelluksen mukaan
- Tuotantoprojekteissa JSX-toteutukset käännetään etukäteen JavaScriptiksi esim. npm:ää käyttämällä, jotta sovelluksen renderöinti saadaan nopeammaksi.

Komponentin renderöinti JSX-merkinnällä:

```
class MovieTitle extends React.Component {
  render() {
    return (
        <p>{this.props.title}</p>
    );
  }
}

React.render(<MovieTitle title="Terminator" />, mountNod
```

Komponentin renderöinti ilman JSX-merkintää:

```
var MovieTitle = React.createClass({displayName: "Movie
    render: function() {
        return React.createElement("p", null, "Hello ",
    }
});

React.render(React.createElement(MovieTitle, {title: "Te
```

Props-objekti ja komponentin saama data

- Reactissa data kulkeutuu käytännössä yhteen suuntaan: vanhemmalta lapselle.
- Nämä ominaisuudet määritellään JSX-merkinnässä komponentin attribuuttien kautta.
- Komponentin sisällä ominaisuuksiin/dataan päästään käsiksi this.props-objektin kautta.

Esimerkki react04-99.html

Havainnot

- Alla olevassa esimerkissä App-komponentille välitetään header- ja text-attribuuttien avulla tekstiä ja App-komponetista number-attribuutin avulla arvottu luku Number-komponentille.

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Web-ohjelmointi: React / props</title>
    <script src="https://unpkg.com/react@16/umd/react.de
    <script src="https://unpkg.com/react-dom@16/umd/reac
    <script src="https://unpkg.com/babel-standalone@6.15
    </head>
    <body>
        <div id="container"></div>
        <script type="text/babel">

class Number extends React.Component {
  render() {
    return (
      <p>Number : {this.props.number}</p>
    );
  }
}

class App extends React.Component {
  render() {
    var number = Math.random();
    return (
      <div>
          <h1>{this.props.header}</h1>
          <p>{this.props.text}</p>
          <Number number = {number}/>
       </div>
    );
  }
}
```

```
34
35  ReactDOM.render(
36      <App header="This is a header" text="This is a text"
37      document.getElementById('container')
38  );
39          </script>
40      </body>
41  </html>
```

Esimerkki: e/09/react04-99.html

this.props.children

- React käyttää this.props.children-objektia komponentin lapsien läpikäyntiin
- Esimerkkinä näytetään komponentin lapset `ol`-listan `li`-elementtien sisältönä
- this.props.children-objektilla voi olla kolme eri vaihtoehtoista arvoa (omissa toteutuksissa tulee olla tarkkana)
  - vain yksi lapsisolmu -> arvo on objekti
  - useampi lapsisolmu -> arvo on taulukko
  - ei lapsisolmuja -> arvo on undefined
- Esimerkkinä näytetään komponentin lapset (useampi lapsisolmu) `ol`-listan `li`-elementtien sisältönä

```
1   <!DOCTYPE html>
2   <html>
3   <head>
4       <meta charset="UTF-8" />
5       <title>react05-99.html</title>
6   </head>
7   <body>
8       <div id="root"></div>
9
10      <script src="https://unpkg.com/react@16/umd/react.de
11      <script src="https://unpkg.com/react-dom@16/umd/reac
12      <script src="https://unpkg.com/babel-standalone@6.15
13
```

```
14      <script type="text/babel">
15
16      class TodoList extends React.Component {
17        render() {
18          return (
19            <ol>
20            {
21                // Läpikäydään komponentin lapset ja luoda
22                // Käytetään lambda-merkittyä funktiota (a
23                React.Children.map(this.props.children, ch
24                    <li>{child}</li>
25                )
26            }
27            </ol>
28          );
29        }
30      }
31
32          ReactDOM.render(
33              <TodoList>
34                  <span>Eka taski</span>
35                  <span>Toka taski</span>
36              </TodoList>,
37              document.getElementById("root")
38          );
39      </script>
40  </body>
41  </html>
```

Esimerkki: e/09/react05-99.html

State-objekti ja komponentin tila

- Aiemmissa esimerkeissä dataa välitettiin komponentilta sen lapsikomponenteille props-objektin avulla
- Komponentin state-objektin avulla voidaan komponentin sisältämää tietoa tallentaa, muuttaa ja esittää käyttöliittymässä (== komponentin tila).
- Käyttöliittymä saadaan päivitettyä kutsumalla this.setState()-metodia, joka kutsuu komponentin render()-metodia.

Esimerkki react06-99.html

Esitään pieni sovellus, jossa painiketta klikkaamalla voidaan laskea klikkausten lukumäärää ja näyttää päivitetty lukumäärä käyttöliittymässä

Havainnot:

- Counter-komponentin constructor-metodi on kirjoitettu näkyviin (rivit 16-20), jotta komponentin alustuksessa vastaanotettu laskurin props.initialCount-ominaisuuden arvo 0 voidaan asettaa alkuarvoksi tilamuuttujalle this.state.count (rivi 18) constructor-metodia kutsutaan ennen kuin komponentti liitetään (ja renderöidään) käyttöliittymään.
- Laskurin arvo ei ole muutettavissa Counter-komponentissa this.props-objektin kautta (rivi 24).
- Käyttäjän klikatessa painiketta kutsutaan komponentin handleClick-metodia, joka kutsuu komponentin `this.setState`-metodia kasvattaen ensin tilamuuttujan `this.state.count` arvoa yhdellä (rivi 25).
- komponentin `this.setState`-metodin kutsuminen aiheuttaa automaattisesti komponentin render-tapahtuman kutusmisen, joten mahdollisesti muuttunut data näkyy käyttöliittymässä
- Rivit 17 ja 19 ovat välttämättömiä, mutta niiden tarkemman merkityksen selvittäminen ei kuulu tämän materiaalin sisältöön

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8" />
5      <title>react06-99.html</title>
6  </head>
7  <body>
8      <div id="root"></div>
9      <script src="https://unpkg.com/react@16/umd/react.de
10     <script src="https://unpkg.com/react-dom@16/umd/reac
11     <script src="https://unpkg.com/babel-standalone@6.15
12     <script type="text/babel">
13
14 // Counter component
15 class Counter extends React.Component {
16   constructor(props) {
17     super(props);
18     this.state = {count: props.initialCount};
19     this.handleClick = this.handleClick.bind(this);
```

```
20      }
21
22      // handle button click event
23      handleClick () {
24          // this.props.initialCount++; <- doesn't work "rea
25          this.setState({count: this.state.count + 1});
26      }
27
28      render() {
29        return (
30          <div>
31              <p>Count : {this.state.count}</p>
32              <button onClick={this.handleClick}>Click!</but
33          </div>
34        );
35      }
36
37  }
38
39  // render
40  ReactDOM.render(
41      <Counter initialCount={0}/>,
42      document.getElementById("root")
43  );
44      </script>
45      </body>
46  </html>
```

Esimerkki: e/09/react06-99.html

Tapahtumien käsittely

- muistuttaa HTML-elementtien tapahtumien käsittelyä DOMissa
- Reactissa tapahtuman nimi tulee kirjoittaa camelCase-muodossa ja JSX-määrittelyssä tapahtuman käsittely määritellään funktiona ei merkkijonona.

button-elementin onclick-tapahtuman käsittely HTML:ssä ja Reactissa

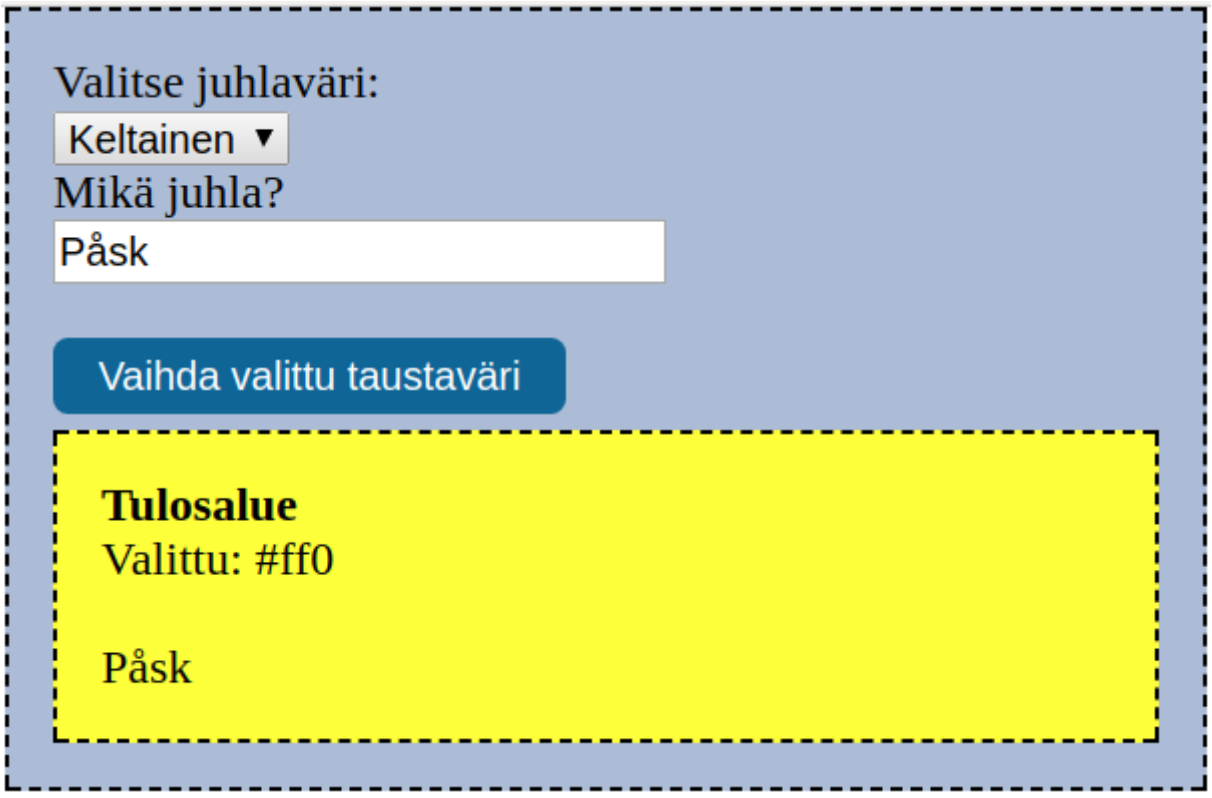
```
// HTML:
<button onclick="doSomething()">Click me!</button>

// React:
<button onClick={doSomething}>Click me!</button>
```

Kaikki Reactin tukemat tapahtumankäsittelijät löydät täältä Supported Events.

Yhteenvetoesimerkki: Tapahtuman käsittelijät, yms.

- Tarkoitettu tutkittavaksi: Kokeile, muuta, ymmärrä!
- Valinnan vaihtaminen pudotusvalikossa tulostaa valitun värikoodin tulosalueelle, mutta ei muuta tulosalueen taustaväriä. Rivit 53, 66 ja 35-37
- "Mikä juhla" -tekstilaatikkoon kirjoitettu teksti päivittyy reaaliajassa kirjoittamisen edetessä tulosalueen alimmalle riville. Rivit 60, 67 ja 38-40
- Pudotusvalikosta valittu taustaväri vaihdetaan tulosalueen taustaväriksi klikkaamalla "Vaihda valittu taustaväri"-painiketta. Rivit 61 ja 41-44
- Huomaa kuinka tulosalueen taustaväriin voi viitata "ref/refs"-viittauksella: Rivit 42 ja 64
- Huomaa että React-komponenteissa elementin class-määrite tulee kirjoittaa muodossa **className** riveillä 61 ja 64.
- Huomaa kuinka React-komponenteihin liitettävät tyylimääritykset asetetaan riveillä 47 ja 64.




```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8" />
    <title>react09.html</title>
    <style>
     body {width: 400px; margin: auto}
    .box {border: 1px dashed #000; padding: 15px; backgr
    .tulosalue {border: 1px dashed #000; padding: 15px;
    input[type="text"] {width: 200px;}
    .nappi { background-color: #069; border-radius: 5px;
                color: #fff; margin-bottom: 5px; margir
                padding: 5px 15px; border: none;
              }
    </style>
</head>
<body>

    <div id="root" class="box"></div>
    <script src="https://unpkg.com/react@16/umd/react.de
    <script src="https://unpkg.com/react-dom@16/umd/reac
    <script src="https://unpkg.com/babel-standalone@6.15

```

```
24      <script type="text/babel">
25
26      class Eventer extends React.Component {
27        constructor(props) {
28          super(props);
29          this.state = {value: '#ff0', textlinevalue: ''};
30          this.handleChange = this.handleChange.bind(this)
31          this.handleTextChange = this.handleTextChange.bi
32          this.handleSubmit = this.handleSubmit.bind(this)
33        }
34
35        handleChange (event) {
36            this.setState({value: event.target.value,});
37        }
38        handleTextChange (event) {
39            this.setState({textlinevalue: event.target.val
40        }
41        handleSubmit (event) {
42          this.refs.colorarea.style.backgroundColor = this
43          event.preventDefault();
44        }
45
46        render() {
47          var divStyle = {backgroundColor: '#ff0'};
48          return (
49            <div>
50              <form onSubmit={this.handleSubmit}>
51                <label>
52                  Valitse juhlaväri:<br />
53                  <select value={this.state.value} onChang
54                    <option value="#ff0">Keltainen</optior
55                    <option value="#f00">Punainen</option>
56                    <option value="#0f0">Vihreä</option>
57                  </select>
58                </label><br />
59                Mikä juhla?<br />
60                <input type="text" onChange={this.handleTe
61                <input className="nappi" type="submit" val
62              </form>
63
```

```
64              <div ref="colorarea" style={divStyle} classN
65                <strong>Tulosalue</strong><br />
66                <span>Valittu: {this.state.value}</span><b
67                <span ref="textline">{this.state.textlinev
68              </div>
69            </div>
70          );
71        }
72      }
73
74  ReactDOM.render(
75      <Eventer />,
76      document.getElementById("root")
77  );
78      </script>
79      </body>
80  </html>
```

Esimerkki: e/09/react08-99.html

Lisäesimerkkejä

Oheiset esimerkit valaisevat samoja Reactin käytön perusasioita kuin yllä on esitetty. Mukana on myös asioita, joita yllä ei ole esitetty. Osin ohjelmakoodit on kirjoitettu hieman vanhemmalla Reactin syntaksilla. Ohjelmakoodeja tutkimalla on kuitenkin mahdollista kirkastaa ja syventää tietämystä Reactin perusteista.

Kokeile, muuta ja yritä ymmärtää seuraavista ainakin esimerkit 1-9

- React-esimerkit 1-6
- React-esimerkit 7-12

Linkkejä

- React
- React Quick Start
- A re-introduction to JavaScript (JS tutorial)
- React Enlightenment

- Virtual Dom
- Learning React.js: Getting Started and Concepts
- React without ES6
- ...